TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky, informatiky a mezioborových inženýrských studií

Studijní program: B2646 Informační technologie

Studijní obor: Informační technologie

Automatizace tvorby grafu v programu Grapher využívající rozhraní k databázi

Automating of Graphs Creation in Grapher Program Using the Interface to the Database

Bakalářská práce

Autor: Michal Běloch

Vedoucí práce: doc. Ing. Milan Hokr, Ph.D.

V Liberci dne 18. 5. 2012

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2011/2012**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal Běloch**
Osobní číslo:	**M08000121**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Automatizace tvorby grafů v programu Grapher využívající rozhraní k databázi**
Zadávající katedra:	**Ústav nových technologií a aplikované informatiky**

Zásady pro vypracování:

1. Seznamte se se softwarem Grapher (Golden Sofwtare) pro tvorbu vědeckých grafů a možnostmi pokročilé konfigurace a programování (Visual basic skripty, rozhraní ODBC).
2. Navrhněte metodiku pro zpracování grafů z průběžně rozšiřované časové řady měření (struktura vstupního souboru, formát časového údaje, automatický update změněného souboru do definovaného vzhledu grafu).
3. Otestujte řešení na různých typech dat a ošetřete jednotnost vzhledu např. pro různě hustá data a více rozsahů os.
4. Předveďte použití rozhranní ODBC pro načítání dat pro grafy z databáze.

Rozsah grafických prací:	**dle potřeby**
Rozsah pracovní zprávy:	**cca 50 stran**
Forma zpracování bakalářské práce:	**tištěná/elektronická**

Seznam odborné literatury:

[1] HOKR M. a kol., Tunel Bedřichov ? charakterizace granitoidů in situ, Závěrečná zpráva, SÚRAO, 2010.

[2] GOLDEN SOFTWARE, Inc., Grapher User?s Guide, 2D & 3D Graphing Software for Scientists, Engineers & Business Professionals, 2011.

Vedoucí bakalářské práce:	**doc. Ing. Milan Hokr, Ph.D.** Ústav nových technologií a aplikované informatiky
Datum zadání bakalářské práce:	**14. října 2011**
Termín odevzdání bakalářské práce:	**18. května 2012**

prof. Ing. Václav Kopecký, CSc.
děkan

Technická univerzita v Liberci - 14 -

prof. Dr. Ing. Jiří Maryška, CSc.
vedoucí ústavu

V Liberci dne 14. října 2011

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím bakalářské práce a konzultantem.

Datum: 18. 5. 2012

Podpis

Poděkování

Na tomto místě bych chtěl poděkovat panu doc. Ing. Milanu Hokrovi, Ph.D., vedoucímu mojí bakalářské práce, za odborné vedení, které mi při psaní práce poskytl.

Dále děkuji své rodině a přátelům za všeobecnou podporu během studia.

Anotace

Automatizace tvorby grafu v programu Grapher využívající rozhraní k databázi

Bakalářská práce se zabývá automatickou tvorbou grafu využívající rozhraní k databázi. Automatizace je vytvářena z dat naměřených v tunelu Bedřichov.

První část této práce seznamuje s programem Grapher, jeho možnostmi připojení k databázi, tvorbě grafů a automatizace. Dále seznamuje s obsahem měření v tunelu Bedřichov a seznamuje s problematikou databází.

V praktické části se zabývá popisem vytvoření automatizace ze stažených dat v programu Grapher. Výsledkem jsou naprogramované skripty, které automatizují tvorbu grafů z naměřených hodnot v tunelu Bedřichov.

Klíčová slova: Automatizace tvorby grafu, program Grapher, tunel Bedřichov, databáze

Annotation

Automating of Graphs Creation in Grapher Program Using the Interface to the Database

This bachelor's thesis deals with automated creation of graphs using interface to database. The automation is created using data collected in the Bedřichov tunnel.

First part of this thesis introduces the Grapher program, its possibilities of connection to the database, creation of graphs and automation. Further it introduces steps of data measurment in the Bedřichov tunnel and explains the database term.

In the practical part it describes creation of automation of acquired data in the Grapher program. The result is programmed scripts that automate the creation of graphs from measurements in the Bedřichov tunnel.

Keywords: Automating of graphs creation, Grapher program, Bedřichov tunnel, database

Obsah

Seznam použitých zkratek

ODBC	Open Database Connectivity
OM	Object Manager
WM	Worksheet Manager
SŘBD	Systém řízení báze dat
SQL	Structured Query Language
API	Application Programming Interface
TUL	Technická univerzita v Liberci
HW	Hardware
GSM	Globální systém pro mobilní komunikaci
TCP/IP	Transmission Control Protocol / Internet Protocol
CSV	Comma-Separated Values
XML	Extensible Markup Language
DSN	Data Source Name
SÚRAO	Správa uložišť radioaktivního odpadu

1 Úvod

Dnes se běžně setkáváme s programy, které uchovávají množství dat v databázi i s programy co tvoří jednoduché grafy. Vedle toho se setkáme se specializovanými programy (pro tvorbu kvalitních grafů), kde ale uživatel pracuje se statickými daty. V mé práci se snažím spojit tyto technologie, což sice některé programy umožňují, ale není to rutinně hromadně rozšířené a vyžadují individuální přizpůsobení uživatele či programátora. Práce vychází z potřeb měření v tunelu Bedřichov prováděného týmy na Fakultě mechatroniky a na Ústavu pro pokročilé technologie, nanomateriály a inovace TUL v rámci jejich výzkumných projektů.

Program, ve kterém budu zpracovávat automatizaci, splňuje potřeby a také navazuje na můj bakalářský projekt, který jsem pojal jako rešerši na téma Grapher, Excel, MatLab a jejich možnosti připojení k databázi [3]. Rešerši jsem zpracovával paralelně s bakalářskou prací. Seznámil jsem s v ní s možnostmi jednotlivých programů v automatizaci a jejich spojením s databází. Již na začátku zpracovávání projektu se mi jevil program Grapher jako vhodný pro automatizaci tvorby grafů a připojení k databázi pro mojí bakalářskou práci.

Na začátku mojí bakalářské práce jsem se naučil pracovat s programem Grapher od firmy (Golden Software) který se používá pro tvorbu vědeckých grafů. S možnostmi tvorby 2D a 3D grafů a jejich vlastnostmi, možnostmi čar, vyplnění grafu, rozsahy os a možností jejich automatizace, dále se podrobněji seznámil s programovacím jazykem WinWrap Basic, který se používá pro programování skriptů.

Cílem bakalářské práce je vytvořit skripty zobrazující grafy z hodnot naměřených čidly a senzory z tunelu Bedřichov. Vytvořit ODBC rozhraní, které se připojí a stáhne z databáze naměřená data. Poté nabídnout možnost zobrazení grafu z jednoho senzoru dle jeho identifikačního čísla. Dále vytvořit skript, který zobrazí více vizuálně odlišených čar v jednom grafu, pro porovnání naměřených hodnot.

2 Program Grapher

2.1 Úvod do programu Grapher

Grapher je sofistikovaný grafický program, který převádí data do jakéhokoliv typu grafu. Distribuovaný firmou Golden Software. Program je určen pro vědce, vývojáře a manažery obchodů, kteří potřebují zobrazit v grafu své údaje. Grapher vytváří více než 50 různých typů grafů. Od 2D, 3D, polárních a speciálních grafů po vrstevnicové a povrchové mapy. Nejnovější verze programu je Grapher 9. Jeho cena je 349 amerických dolarů [5].

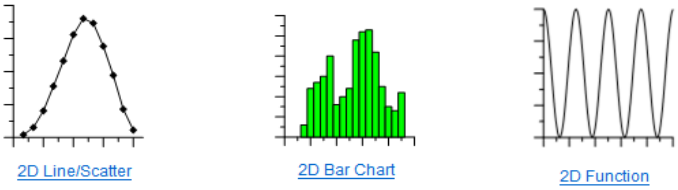
Nástroj pro automatizaci tvorby grafů, které Grapher používá, je takzvaný Scripter, se kterým se seznámíme v kapitole 2.4.

2.2 Typy grafů

Grapher vytváří několik unikátních 2D nebo 3D typů. Příklady některých typů jsou uvedeny v podkapitolách 2.2.1 až 2.2.4.

2.2.1 2D XY grafy

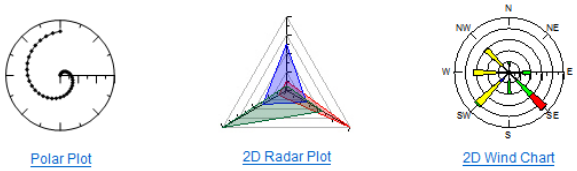
2D XY jsou dvourozměrné plochy, které zobrazují data jako čáry, body nebo bary na 2 osách. Všechny vlastnosti grafu jsou upravovatelné včetně os zobrazující symboly a čáry a u barů šířku pruhů. Tyto grafy budu pro automatizaci využívat nejvíce, z hlediska jejich dobře upravitelných vlastností.



Obr. 1 - 2D grafy

2.2.2 Polární grafy

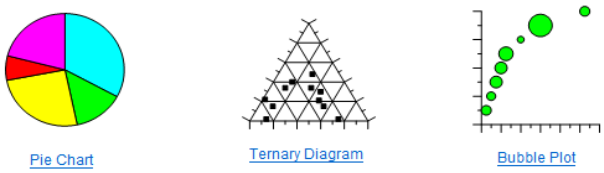
Polární grafy zobrazují data jako čáry, body a bary na polárních osách. Údaje jsou definovány úhly a vzdáleností od středu grafu. Existují různé možnosti pro každý typ grafu, včetně nastavení vlastností os, vlastností čar, vlastností symbolů a barových vlastností.



Obr. 2 - Polární grafy

2.2.3 Speciální grafy

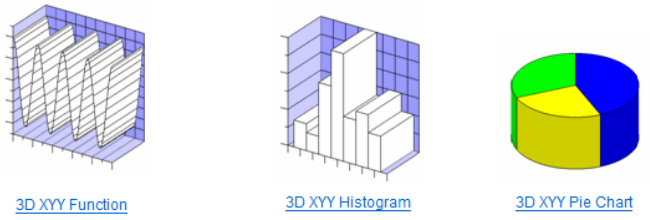
Speciální grafy obsahují mnoho různých typů grafů. Různé vlastnosti, včetně vlastností os, vlastností čar, vlastnosti šířky barů a počet řezů.



Obr. 3 – Speciální grafy

2.2.4 3D XYY grafy

3D XYY grafy zobrazují dvourozměrné údaje ve třech rozměrech. To se provede nastavením třetího rozměru grafu do šířky. Všechny vlastnosti, včetně vlastností čar, vlastností výplně, vlastností os a šířkou grafu jsou nastavitelné.



Obr. 4 - 3D XYY grafy

2.3 Uživatelské rozhraní programu Grapher

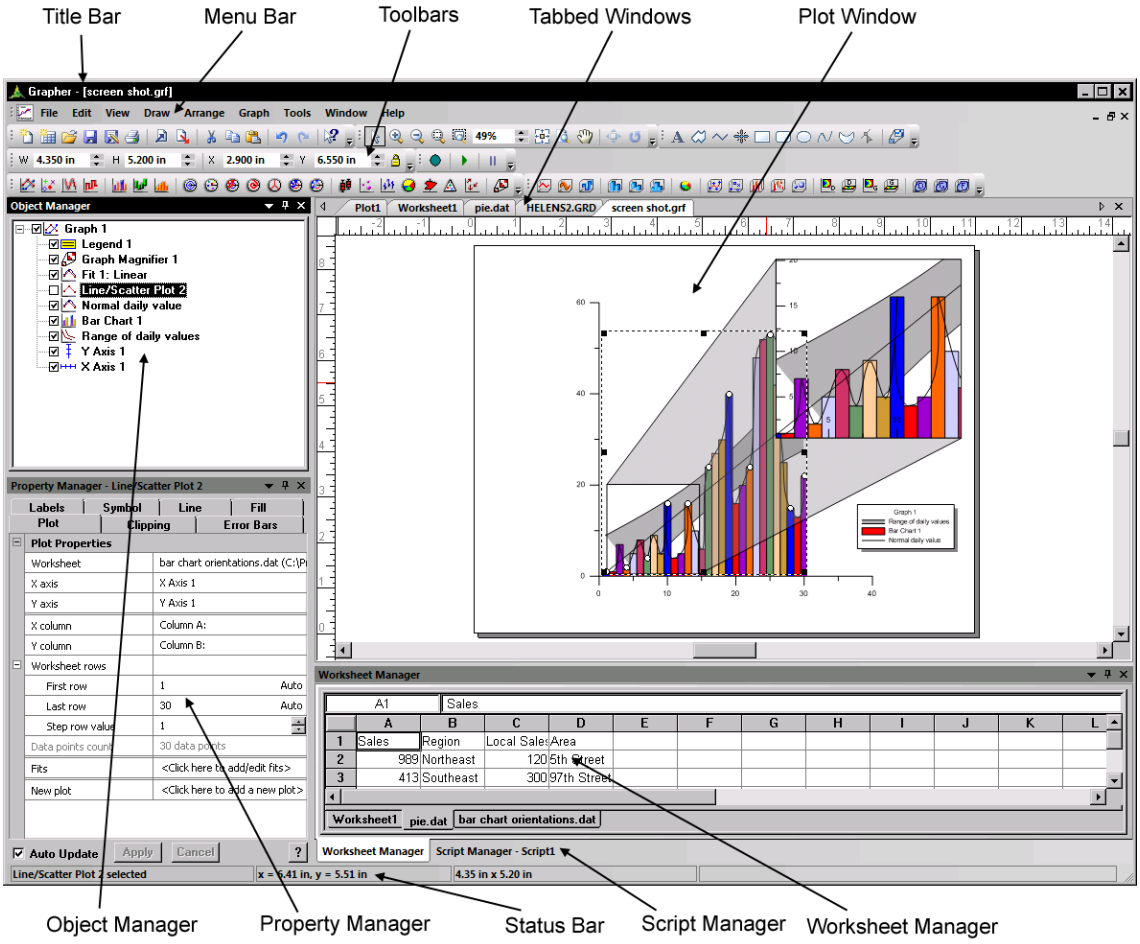
Uživatelské rozhraní programu Grapher se skládá z Title Bar, Menu Bar, Toolbars, Tabbed Windows, Object Manager a Status Bar. Grapher obsahuje 4 typy okenních dokumentů: Plot Window, Worksheet Manager, Excel worksheet Windows a Grid Windows.

Grafy a mapy se zobrazují a mohou být upraveny v Plot Window.

Worksheet Windows zobrazuje, upravuje, převádí a ukládá data tabulkového formátu.

Excel worksheet Windows umožňuje otevřít původní Excel okno v programu Grapher.

Jednotlivé komponenty popisuji v **Tabulka 1**



Obr. 5 - Uživatelské rozhraní programu Grapher

Tabulka 1 - Funkce komponent programu Grapher

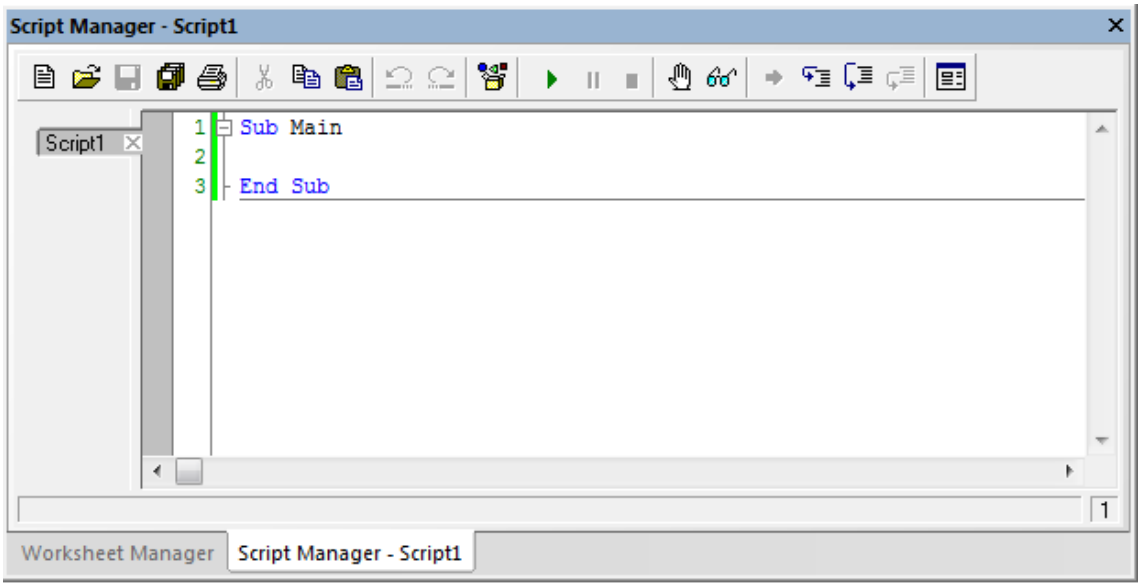
Jméno komponenty	**Funkce komponenty**
Title Bar	Záhlaví obsahuje název programu plus uložený název, pokud existuje. Hvězdička za názvem souboru označuje, že soubor byl upraven před posledním uložením.
Menu bar	Menu lišta obsahuje používané příkazy ve spuštěném programu Grapher
Toolbars	Panel nástrojů obsahuje tlačítka nástrojů programu Grapher, které jsou zkratky menu příkazů.
Tabbed Window	Vícenásobný plot Windows, worksheet Windows, Excel Windows a Grid Windows se zobrazují jako záložky. Kliknutím na záložku se zobrazí obsah okna.
Object Manager	Object Manager obsahuje hierarchický seznam všech objektů obsažených v Plot Window. Tyto objekty mohou být vybrány, přidány, uspořádány upraveny a přejmenovány v Object Manager. OM je původně ukotven na levé straně nad Property Manager. Změny provedené v OM se okamžitě projeví v Plot Window.
Property Manager	Property Manager umožňuje upravovat některou z vlastností vybraného objektu. Více objektů může být naráz upraveno vybráním všech objektů a změnou sdílených vlastností. Změny provedené v Property Manager se ihned projeví v Plot Window.
Worksheet Manager	Worksheet Manager obsahuje přehled o všech datech nahraných do programu Grapher. Úpravy provedené ve WM se automaticky promítnou v grafu.
Script Manager	Správce skriptů kontroluje nahrané a běžící skripty v programu Grapher.
Status Bar	Stavový řádek zobrazuje informace o činnostech v programu Grapher. Stavový řádek je rozdělen do tří částí, které obsahují informace o vybraných příkazech, objektech nebo pozicích.

2.4 Scripter

Scripter, jako součást programu Grapher, je vhodný pro vytváření, editaci a provádění skriptů, které automatizují operace programu Grapher. S využitím skriptů mohou být jednoduché každodenní, ale i komplexní úkoly prováděny opakovaně bez přímé interakce [1].

Script recorder zaznamenává všechny příkazy prováděné v programu Grapher. Když je skript spuštěn, Grapher provede zaznamenané kroky. Což je vhodné pro uživatele, kteří často provádí stejné úkoly, ale nejsou zběhlí v automatizaci, pro pokročilé uživatele, kteří mají problémy se syntaxí nebo ji pouze nechtějí celou zadávat ručně.

V mé bakalářské práci budu ve Scripteru vytvářet a pracovat se skripty pro připojení k databázi a budu ho využívat pro automatizaci tvorby grafů.



Obr. 6 – Grafické rozhraní Scripteru

Scripter si v programu Grapher odkryjeme v liště menu v možnosti View, Managers a zaškrtneme Script Manager.

Vrchní část je zobrazený Toolbar, kde jsou možnosti, které vytvoří nový skript, otevřou uložený skript, tisknou, spustí skript. V levé části jsou aktuálně otevřené skripty. Zbylá část, je pracovní okno, ve kterém se píší skripty.

Scripter je napsán ve WinWrap Basic Leanguage a k programování skriptů používá Visual Basic for Applications(TM) kompatibility.

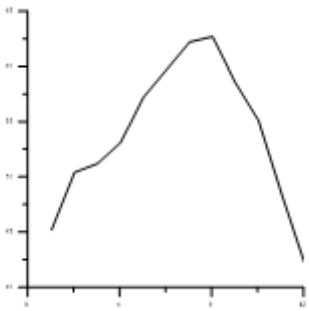
V programu Grapher, Scripter používá knihovnu obsahující data o všech typech grafů, worksheetu, možnostech ukládání, načítání grafů a vlastnostech jednotlivých os grafů. Knihovna není přístupná odjinud, než z programu Grapher.

2.4.1 Ukázka a popis skriptu pro automatickou tvorbu grafu:

```
Sub Main
'Vytvoří Grapher jako objekt
Dim Grapher As Object
'Spustí Grapher
Set Grapher = CreateObject("Grapher.Application")
Grapher.Visible = True
'Vytvoří nové okno dokumentu
Set Plot1 = Grapher.Documents.Add(grfPlotDoc)
'Vytvoří osy grafu z datového souboru
Plot1.Shapes.AddLinePlotGraph("C:\Program      Files\Golden
Software\Grapher 9\Samples\Tutorial.dat",1,2)
'Definuje objekt ukazatele
Set Graph1 = Plot1.Shapes.Item(1)
'Definenuje objekt ukazatele
Set LineScatterPlot1 = Graph1.Plots.Item(1)
'Vybere objekt
LineScatterPlot1.Select
End Sub
```

Výsledný graf:

Je jednoduchý vytvořený 2D Plot, kde je pouze Xová a Ynová osa a v grafu je zobrazeno pouze 10 bodů.



Obr. 7 - Ukázkově vytvořený graf

3 Teorie databází

Databáze je určitá uspořádaná množina informací uložená na nějakém paměťovém médiu [4]. V širším smyslu jsou součástí databáze i softwarové prostředky, které umožňují manipulaci s uloženými daty a přístup k nim. Tento software se v české odborné literatuře nazývá systém řízení báze dat (SŘBD).

3.1 Historie

Předchůdce databází byly papírové kartotéky. Správa takový kartoték byla v mnohém podobná správě dnešních databází.

Velký impulz pro rozvoj databází byl vývoj počítačů v 50 letech 20 století. Ukázalo se, že původně univerzální používání strojového kódu procesorů je (nejen) pro databázové úlohy neefektivní, a proto se objevil požadavek na vyšší jazyk pro zpracování dat.

To mělo za následek vytvoření vyšších programovacích jazyků. V roce 1962 jazyk COBOL a v roce 1965 jazyk PL/1.

V roce 1970 začínají zveřejněním článku E. F. Codda první **relační databáze**, které pohlížejí na data jako na tabulky. Kolem roku 1974 se vyvíjí první verze dotazovacího jazyka SQL.

V 90. letech 20. století se začínaly objevovat první **objektově** orientované databáze, jejichž filozofie byla přebírána z objektově orientovaných jazyků.

Pojem „databáze“ je často zjednodušován na to, co je ve skutečnosti databázový systém (databázový stroj) nebo též systém řízení báze dat. Ten neobsahuje pouze **tabulky** – ty jsou jedny z mnoha tzv. databázových objektů (někdy též databázových entit). Pokročilejší databázové systémy obsahují například:

- **pohledy** neboli **views** – SQL příkazy, pojmenované a uložené v databázovém systému. Lze z nich vybírat (aplikovat na ně příkaz SELECT) jako na ostatní tabulky.
- **indexy** neboli **klíče** pro každou tabulku. Klíče jsou definovány nad jednotlivými sloupci tabulek (jeden klíč jich může zahrnovat i více) a jejich funkce je vést si v tabulkách rychlé LUT (*look-up tables* – „pořadníky“) na sloupce, nad nimiž byly

definovány, vyloučit duplicitu v záznamech nebo zajišťovat fulltextové vyhledávání.

- **spouště** neboli **triggery** – mechanismus mezi řádky dvou tabulek, který se v databázovém systému dá definovat jako jeden z několika úkonů, který se vyvolá po změně nebo smazání rodičovské tabulky.
- **procesy** – databázové stroje umí podat přehled o procesech, které jejich služeb aktuálně využívají.

3.2 Relační databáze

Základním konstruktorem relačních databází jsou relace (databázové tabulky), což jsou dvourozměrné struktury tvořené záhlavím a tělem. Jejich sloupce se nazývají atributy, řádky tabulky jsou pak záznamy. Atributy mají určen svůj konkrétní datový typ a doménu, což je množina přípustných hodnot daného atributu. Řádek je řezem přes sloupce tabulky a slouží k vlastnímu uložení dat [9].

Terminologie:

Kandidátní klíč - Kandidátní klíč je atribut nebo skupina atributů, které jednoznačně identifikují záznam v relační tabulce. Kandidátní klíč se může stát primárním klíčem; ty které se primárním klíčem nestanou jsou označovány jako alternativní klíče.

Primární klíč - je jednoznačný identifikátor záznamu, řádku tabulky. Primárním klíčem může být jediný sloupec či kombinace více sloupců tak, aby byla zaručena jeho jednoznačnost. Pole klíče musí obsahovat hodnotu, tzn. nesmí se zde vyskytovat nedefinovaná prázdná hodnota NULL.

Cizí klíč - Dalším důležitým pojmem jsou nevlastní/cizí klíče. Slouží pro vyjádření vztahů, relací, mezi databázovými tabulkami. Jedná se o pole či skupinu polí, která nám umožní identifikovat, které záznamy z různých tabulek spolu navzájem souvisí.

Vztahy mezi tabulkami:

Vztahy (relationships) slouží ke svázání dat, která spolu souvisejí a jsou umístěny v různých databázových tabulkách. Každý vztah je charakterizován třemi základními vlastnostmi:

- stupněm
- kardinalitou
- volitelností účasti

Stupeň vztahu

1. Unární vztah - relace je spojena sama se sebou. Typickým příkladem je vztah Zaměstnanec - Nadřízený, kdy nadřízený je také jedním ze zaměstnanců a může mít také nadřízeného. Vztah se realizuje vložením primárního klíče relace Zaměstnanec ve formě cizího klíče opět do relace Zaměstnanec.
2. Binární vztah - klasický vztah mezi dvěma relacemi.
3. Ternární vztah - jedná se o vztah mezi třemi relacemi najednou.
4. Enární vztah - jedná se vztah mezi n-relacemi zároveň.

Ternární a enární vztahy se nesnadno modelují a v praxi se objevují velice zřídka.

Kardinalita vztahu

1. mezi daty v tabulkách není žádná spojitost, proto nedefinujeme žádný vztah.
2. 1:1 používáme, pokud záznamu odpovídá právě jeden záznam v jiné databázové tabulce a naopak. Takovýto vztah je používán pouze ojediněle, protože většinou není pádný důvod, proč takovéto záznamy neumístit do jedné databázové tabulky. Jedno z mála využití je zpřehlednění rozsáhlých tabulek. Jako ilustraci je možné použít vztah řidič - automobil. V jednu chvíli (diskrétní časový okamžik) řídí jedno auto právě jeden řidič a zároveň jedno auto je řízeno právě jedním řidičem.
3. 1:N přiřazuje jednomu záznamu více záznamů z jiné tabulky. Jedná se o nejpoužívanější typ relace, jelikož odpovídá mnoha situacím v reálném životě. Jako reálný příklad může posloužit vztah autobus - cestující. V jednu chvíli cestující jede právě jedním autobusem a v jednom autobuse může zároveň cestovat více cestujících.

4. M:N je méně častým. Umožňuje několika záznamům z jedné tabulky přiřadit několik záznamů z tabulky druhé. V databázové praxi bývá tento vztah z praktických důvodů nejčastěji realizován kombinací dvou vztahů 1:N a 1:M, které ukazují do pomocné tabulky složené z kombinace obou použitých klíčů (třetí resp. tzv. vazební tabulka). Příkladem z reálného života by mohl být vztah výrobek - vlastnost. Výrobek může mít více vlastností a jednu vlastnost může mít více výrobků. V reálném životě nicméně existuje velké množství vztahů M : N, mimo jiné také proto, že často existuje praktická potřeba zachovávat i údaje o historii těchto vztahů z časového hlediska (jeden řidič v delším časovém období řídí více rozličných aut a jedno auto v delším časovém období může mít více různých řidičů).

Volitelnost účasti ve vztahu

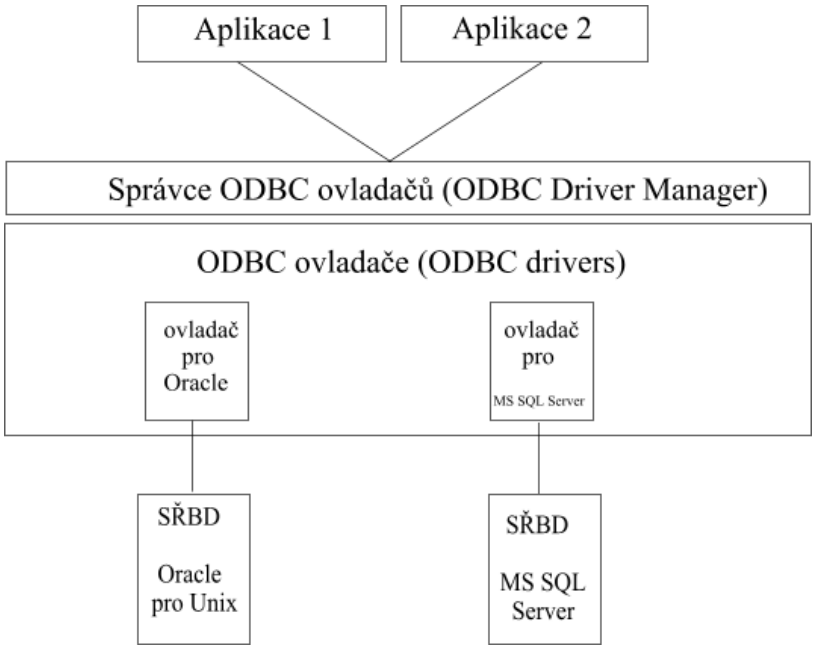
Volitelnost účasti ve vztahu vyjadřuje, zda je účast relace ve vztahu povinná nebo volitelná.

3.3 Open Database Connectivity

ODBC (Open Database Connectivity) – je programovací rozhraní, které aplikacím umožňuje přístup k datům v systémech řízení bází dat, které jako standard pro přístup k datům používají jazyk SQL (Structured Query Language) [8].

Použitím API ODBC mohou aplikace přistupovat k datům, nezávisle na tom, jakým systémem pro řízení báze dat (dále jen SŘBD resp. DBMS) jsou tato data spravována a to i přesto, že každý SŘBD používá pro uložení dat jiný formát a jiné programové rozhraní. Zavedení ODBC, jakožto standardu, pro přístup k datům se ukázalo být velmi prozíravým činem, který přinesl mnoho výhod nejen programátorům (resp. koncovým uživatelům), ale i samotným výrobcům systémů pro řízení báze dat. Přestože za tvorbou ODBC stojí společnost Microsoft, tak použití ODBC není nijak vázáno pouze na platformu Win32.

Architekturu ODBC vidíte na Obr. 8



Obr. 8 – Schéma ODBC

Model struktury ODBC se dá znázornit pomocí čtyř vrstev.

- **V první nejvrchnější vrstvě** se nachází samotná aplikace. Ta v případě, že potřebuje data, provede volání ODBC funkcí (ve formě SQL dotazu).
- **Druhou vrstvou** je tzv. „Správce ODBC ovladačů“ (ODBC Driver Manager). Úkolem správce ovladačů je zajistit propojení mezi aplikací a příslušným ODBC ovladačem (ODBC ovladače tvoří třetí vrstvu modelu, podrobněji viz dále). Jakmile aplikace potřebuje data, správce ovladačů vyhledá a nahraje příslušný ovladač. (ve formě DLL knihovny). Správce ovladačů také zjistí jaké konkrétní funkce jsou podporovány jednotlivými ovladači a uschová si jejich adresy v paměti do tabulky. V případě, že aplikace volá konkrétní funkci, správce souborů zjistí, ke kterému ovladači funkce patří a zavolá ji. Tímto způsobem může být prováděn souběžný přístup k více ovladačům, což se hodí v případě programování aplikací přistupujících souběžně k několika zdrojům dat.
- **Třetí vrstvou** zde již zmíněnou vrstvou jsou ODBC ovladače. Ty provedou zpracování volané ODBC funkce, přeložení požadavku do SQL pro příslušný SŘBD (DBMS) a jeho následné poslání.
- **Poslední vrstvou** je SŘBD, který provede zpracování operace požadované ODBC ovladačem a výsledky této operací mu vrátí.

Hlavní a zásadní výhodou plynoucí z použití ODBC je zjednodušení přístupu do databáze. ODBC zavádí abstrakci nazvanou „data source“ (datový zdroj). Pod pojmem „data source“ si lze představit nějaké smysluplné symbolické jméno pro zdroj dat např. platba, inventář apod. ODBC pak toto jméno přiřazuje přes konkrétní ovladač, síťový software, jméno serveru nebo adresu do SŘBD. Přitom je úplně jedno, zda se zdroj dat nachází v lokální databázi nebo někde na síti. Programátor je tímto zbaven povinnosti zabývat se jakýmikoliv síťovými záležitostmi.

4 Tunel Bedřichov

4.1 Přírodní podmínky

Lokalita Bedřichovský tunel se nachází v Jizerských horách v severních Čechách, mezi obcemi Bedřichov a Janov a nachází se v něm vodovodní potrubí.

Tunel byl ražen v letech 1979 – 1981, měří 2 600 m, průměr je 3,3 m a vodovodní potrubí má průměr 80 cm. Výzkum zde začal v letech 2002 – 2003 zakázkami od SÚRAO pro ČGS.

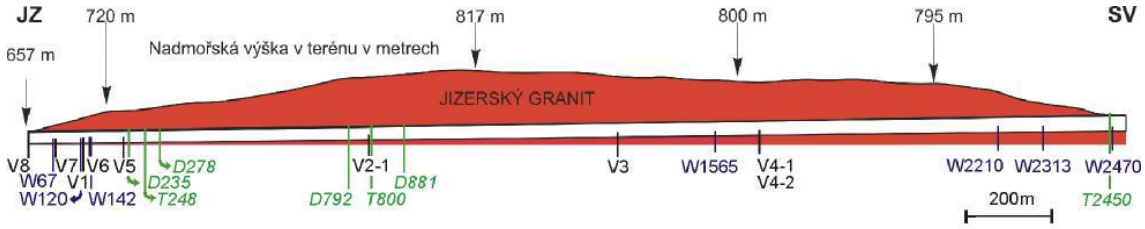
- 2002 – 2003 geologická a strukturní charakteristika granitoidů z tunelu Bedřichov v Jizerských horách
- 2004 – 2005 geologická a strukturní charakteristika granitoidů z vodárenských tunelů v Jizerských horách.
- 2006 – 2008 studium dynamiky puklinové sítě granitoidů ve vodárenském tunelu Bedřichov v Jizerských horách.
- Od roku 2009 převzala práce v tunelu Bedřichov TUL.

4.2 Obsah měření

Práce na úkolu „Projekt Tunel 2011“ započaly dne 24. 6. 2011. V tunelu je budován systém pro automatický sběr dat [6]. Data zaznamenané senzory uvnitř tunelu jsou přenášena na vzdálený server, kde se ukládají do databáze. V tunelu se provádějí seizmická, teplotní a odporové měření [7].

4.2.1 Monitorování

Do monitorovacího programu patří sada pravidelně monitorovaných pramenů. Místa monitorovaných podzemních pramenů jsou označena na stěně tunelu A písmenem V a pořadovým číslem. Místa nově monitorovaných pramenů byla označena písmenem W a pozicí pramenu (v metrech od portálu tunelu). Všechny pravidelně monitorované prameny jsou uvedeny níže a jejich pozice je zobrazena na Obr. 9



Obr. 9 - Schéma tunelu Bedřichov.

Přehled měřených veličin, způsob frekvence měření a odběru vzorků na jednotlivých pramenech přidávám v Příloze A – Seznam použitých měřících zařízení v Tunelu Bedřichov.

4.2.2 Přenos dat z tunelu Bedřichov

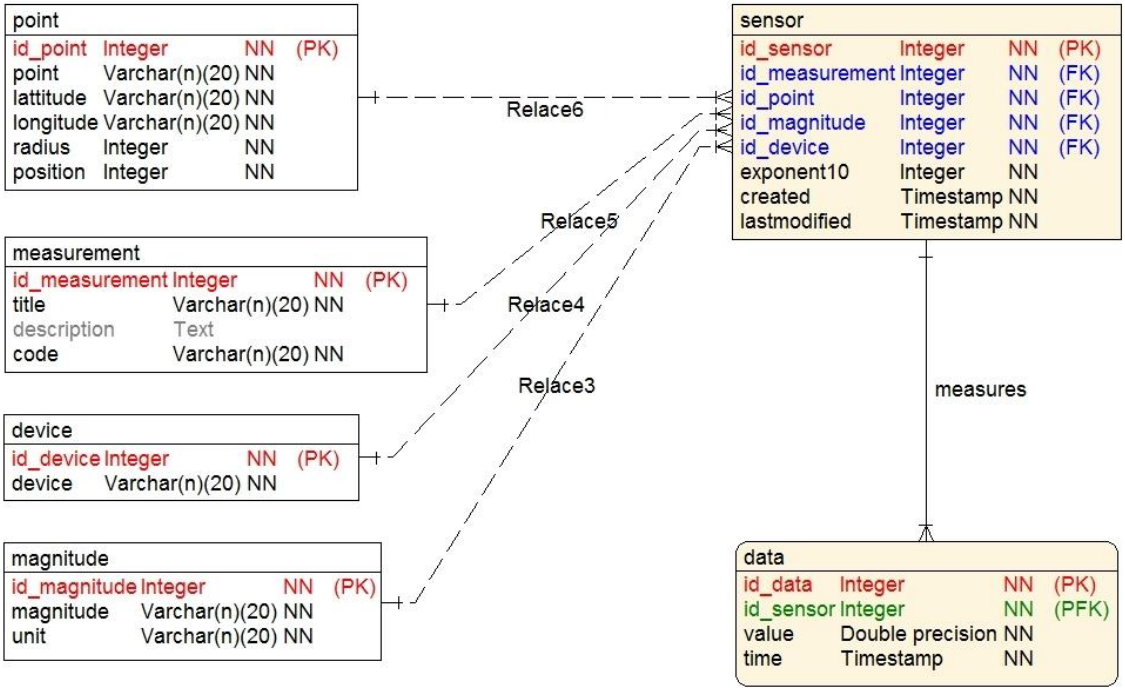
Spočívá v odesílání dat měřených v určité lokalitě na databázový server, kde proběhne jejich uložení. Jednotlivé měřící přístroje v lokalitě jsou propojeny sběrnicí RS485 k zařízení s HW modulem TC65i. Zařízení sbírá data a pak odesílá na server [10].

Skládá se z:

- Senzorů umístěných v tunelu
- Řídícím a komunikačním systémem s nízkonapěťovým microcontrolerem a GSM modulem u vstupu do tunelu komunikujícím s umístěnými senzory a vysílající shromážděná data přes bezdrátovou síť
- Aplikace na serveru (zvaná Remote Data Receiver) zodpovědná za:
    - Přijímaní TCP/IP paketů
    - Získávání naměřených dat uložených ve EPSNet paketech (originální pakety se užívají pro komunikaci v tunelu)
    - Dekódovat data z EPSNet paketů a ukládat je do struktury relační databáze
- Databázový server věnující se ukládání naměřených dat do struktury navržené relační databáze

4.3 Databáze k Tunelu Bedřichov

Data monitorované v Tunelu Bedřichov, se ukládají na databázový server. Tento databázový server je umístěn na portále [2]. Díky tomu je možné získat přehled všech senzorů a naměřených dat. Pro účely mojí bakalářské práce poslouží databáze k Tunelu Bedřichov jako předloha pro vytvoření vlastní databáze. Schéma databáze je na Obr. 10



Obr. 10 – Schéma navržené databáze k tunelu Bedřichov

Popis atributů v databázových tabulkách:

Tabulka Data: Obsahuje naměřenou hodnotu, senzor, na kterém byla naměřena a čas.

Tabulka Senzor: Obsahuje Název měření, Bod měření, přístroj a veličinu

Tabulka Bod: Obsahuje název bodu, latitude, longitude, radius a pozici.

Tabulka Měření: Obsahuje Název a popis měření.

Tabulka Přístroje: Obsahuje název přístroje a číslo přístroje.

Tabulka Veličina: Obsahuje Název veličiny a její jednotku.

4.3.1 Vztahy mezi tabulkami v databázi

V tabulce data je obsažen id_sensor, což je cizí klíč. Touto hodnotu se přiřazuje senzor, kterým byla data naměřena.

Příklad vysvětlení:

Tabulka 2 - Tabulka Bod bude obsahovat všechny body, kde jsou nainstalované zařízení a na nich konkrétní senzory.

Id_point	point	latitude	longitude	radius	position
1	Bod měření 1	10	80	30	1520
2	Bod 2	30	50	40	1360

Tabulka 3 - Tabulka Zařízení obsahuje seznam použitých senzorů v dole. Pojem Device zde označuje senzor, který provádí měření.

Id_device	Device_cs
1	Senzor teploty
2	Senzor průtoku
3	Hladinoměr

Tabulka 4 - Tabulka Veličina obsahuje veličiny a jejich jednotky, které se používají při měření v dole.

Id_magnitude	Magnitude_cs	Unit_cs
1	Metry krychlové	m3
2	Metry čtvereční	m2
3	Teplota	°C

Tabulka 5 - Tabulka Měření obsahuje názvy a popis měření, která se doposud prováděla v dole.

Id_measurement	Title_cs	description	Code_cs
5	Import dat z obecných zdrojů	Data, která přichází od zařízení nepodporující komunikační protokol tunelu	ID
10	Water log		WT1

Tabulka 6 - Tabulka Senzor.

Id_sensor	Id_measurement	Id_point	Id_magnitude	Id_device	exponent	created	lastmodified
1	5	1	2	1	-	-	-
2	10	2	1	2	-	-	-

Tabulka Sensor obsahuje 4 cizí klíče. Id_magnitude, přiřazuje veličinu z tabulky Veličina k naměřené hodnotě na senzoru, id_device přiřazuje zařízení z tabulky

Zařízení, id_measurement přiřadí měření z tabulky Měření a id_point přiřazuje konkrétní bod z tabulky Bod.

Tabulka data může obsahovat vícekrát hodnotu od stejného senzoru.

V tabulce Senzor bude odkazovat na konkrétní bod a zařízení, na kterém byla data naměřena, bude obsahovat název měření a v jaké veličině, byla data přijata.

Tabulka 7 - Výsledná tabulka po spojení

Id_sensor	Title_cs	point	Magnitude_cs	Device_cs	exponent	created	lastmodified
1	Import dat z obecných zdrojů	Bod měření 1	m2	Senzor teploty	-	-	-
2	Water log	Bod 2	m3	Senzor průtoku	-	-	-

5 Zpracování vlastního řešení

Na začátku tvorby mojí bakalářské práce jsem měl přístup k webu Tunelu Bedřichov [2], kde funguje vytvořená databáze, do které se ukládají všechna data zaznamenaná senzory a čidly umístěnými v tunelu.

Databázi spravuje Ing. Roman Špánek Ph.D. Webové rozhraní mi nabízelo dvě možnosti exportu naměřených dat, do formátu CSV a formátu XML.

Jelikož program Grapher nedisponuje možností připojit se do databáze na serveru, kde běží webové rozhraní, rozhodl jsem se vytvořit si vlastní podobnou databázi, kterou budu spravovat. Pro vytvoření a správu databáze jsem se rozhodl pro program Microsoft Access 2007.

Jednou z možností programu Access je import dat formátu XML a možnost vytvořenou databázi uložit ve formátu (*.mdb), což je vhodné pro program Grapher.

Během tvorby bakalářské práce jsem se setkal s nekompatibilitou softwaru. Práci jsem vypracovával na počítači s nainstalovaným 64bitovým systémem a s nainstalovanou 64bitovou verzí programu Grapher. Bohužel ODBC rozhraní mi na 64bitovém systému nedovolilo spravovat (*.mdb) zdroj dat. Toto omezení mi nedovolilo načítat data z databáze. Proto jsem musel použít 32bitový operační systém, 32bitovou verzi programu Grapher. V takovéto kombinaci jsem už žádné problémy, s načítáním dat z databáze pomocí ODBC, neměl.

5.1 Vytvoření databáze

Databázi jsem vytvořil v programu Microsoft Access 2007. Databáze je vytvořená podle vzoru z kapitoly 4.3 a obsahuje 6 tabulek propojených relacemi.

Tabulka 8 až Tabulka 13 popisuje přiřazený datový typ k atributům v jednotlivých databázových tabulkách. Přidávám i popis k atributům databázových tabulek

Tabulka 8 – Databázová tabulka Bod

Název pole	Datový typ	Popis
ID_point	Automatické číslo	PK (privátní klíč)
Point	Text	Název bodu
Latitude	Text	Zeměpisná šířka
Longitude	Text	Zeměpisná délka
radius	Číslo	radius
position	Číslo	pozice

Tabulka 9 – Databázová tabulka Device

Název pole	Datový typ	Popis
ID_device	Automatické číslo	PK (privátní klíč)
Device_cs	Text	Český popis zařízení

Tabulka 10 – Databázová tabulka Magnitude

Název pole	Datový typ	Popis
ID_magnitude	Automatické číslo	PK (privátní klíč)
Magnitude_cs	Text	Český popis veličiny
Unit_cs	Text	Český popis jednotky

Tabulka 11 – Databázová tabulka Measurement

Název pole	Datový typ	Popis
ID_measurement	Automatické číslo	PK (privátní klíč)
title_cs	Text	Český popis měření
code_cs	Text	

Tabulka 12 – Databázová tabulka Senzor

Název pole	Datový typ	Popis
ID_senzor	Automatické číslo	PK (privátní klíč)
ID_localname	Text	Pracovní označení senzoru
ID_measurement	Číslo	FK (measurement)
ID_point	Číslo	FK (point)
ID_magnitude	Číslo	FK (magnitude)
ID_device	Číslo	FK (device)
exponent10	Číslo	Hodnota exponentu
datasize	Číslo	Velikost dat
created	Datum a čas	Datum, kdy byl vytvořen
lastmodified	Datum a čas	Datum poslední úpravy

Tabulka 13 – Databázová tabulka Data

Název pole	Datový typ	Popis
ID_data	Automatické číslo	PK (privátní klíč)
ID_senzor	Číslo	FK (senzor)
Hodnota	Číslo	
time	Datum a čas	

Zde si lze všimnout jednoho rozdílu mezi původní a mnou vytvořenou databází. Rozdíl je v atributu ID_localname, která v databázi určuje pracovní označení senzoru. V původní databázi tento atribut schází.

Použité datové typy:

- *Automatické číslo* – Je to privátní klíč (jedinečná hodnota) hodnota začínající 1. až po počet záznamů v tabulce. Hodnota se v záznamech nemůže opakovat.
- *Číslo* – V Microsoft Access označeno jako dvojitá přesnost na 15 desetinných míst (float)
- *Text* – počet znaků, které se do pole dají vepsat, je 255.
- *Datum a čas* – obecné datum ve formátu 1.12.2012 19:45:22

5.1.1 Vkládání dat do tabulky

Pro naplnění záznamů v tabulkách používám dva způsoby:

1. **Ruční**

   Vytvářím nové záznamy, ve vizuálním prostředí Accessu, vkládáním hodnot do jednotlivých řádků databázové tabulky. Při vložení záznamu do jakéhokoliv řádku databázové tabulky se automaticky vytvoří Automatické číslo ve sloupci ID_Data. Při vkládání záznamů do sloupce Hodnota, je nutné dbát na formát, ve kterém záznamy vkládám. Jelikož desetinná čárka se v Česku píše takto: 17,1265423214.

   Zařízení, které měří hodnoty v Tunelu Bedřichov, vytvářejí formát 17.1265423214, což samozřejmě nemohu do Accessu, do sloupce Hodnota (dle nastavení Accessu v Česku), zadat.

   Část mojí bakalářské práce testuji na záznamech takto vytvořených. Šlo o vyladění napsaných skriptů v programu Grapher, protože vím, jaké záznamy jsou v databázi a jaké mohu čekat výsledné grafy.

2. **Import ze souboru XML**

   Souborem XML, lze zaplnit tabulku „Data“, ze které se v programu Grapher načítají hodnoty pro vytvoření grafu. Pro načtení souboru XML se v programu

Access musí přepnout do záložky Externí data a poté vybrat tlačítko „Soubor XML“. Dojde k zobrazení formuláře, ve kterém vyberu cestu k souboru. Po načtení souboru se musí nastavit, jak a jaké hodnoty chceme do tabulky Data importovat. Můžeme vytvořit novou tabulku „Data1“ nebo připojit k existující tabulce.

Z důvodu formátu číselných hodnot vkládaných do databázové tabulky, popsané v kapitole o ručním vkládání dat, je vhodné data uložit do nové tabulky „Data1“. Všechny sloupce se nastaví na datový typ Text. Nejjednodušší způsob jak přenést potřebné data do tabulky Data je, že ve sloupci Hodnota (ve vytvořené tabulce z importu) nahradíme všechny tečky za čárky a záznamy potom přeneseme do tabulky „Data“.

Při spuštění skriptu pro porovnání čar z více senzorů, se v první části skript připojí do tabulky „Data1“ a získá informace o jednotlivých senzorech.

Hodnoty pro vytvoření čar pak skript používá z tabulky „Data“

V době, kdy jsem čerpal data z XML souborů, byla databáze uložena na serveru v Praze Ústavu informatiky Akademie věd ČR ve spolupráci s TUL. Nyní se databáze přesunula do Liberce, ale bohužel ještě nefunguje export do XML souboru, takže svoji práci nemohu otestovat na nově vytvořených datech z Tunelu Bedřichov.

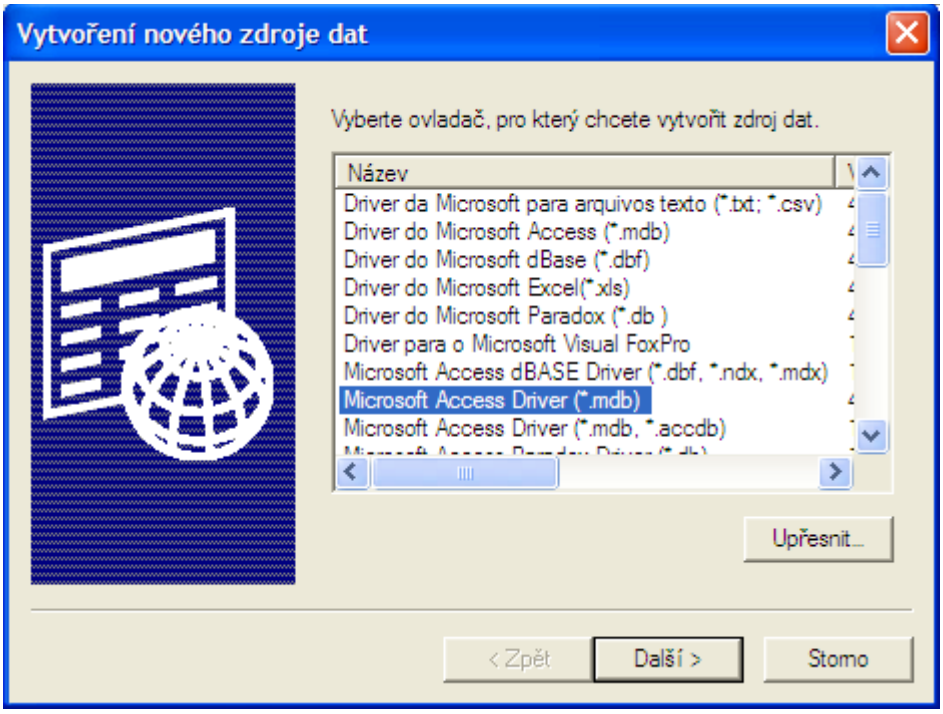
5.1.2 Uložení vytvořené databáze

Hotovou databázi ukládám na absolutní cestu C:\Grapher\Tunel Bedřichov.mdb ve formátu „databáze aplikace Access 2002 – 2003“. Vybral jsem si tento formát kvůli konektivitě s ODBC.

Program Grapher při načítání uložené databáze rozeznává datové typy přiřazené jednotlivým sloupcům. Tudíž rozezná datový typ Datum a čas a samozřejmě Číslo. Jinak nastavené datové typy ve sloupcích vedou ke špatnému zobrazení hodnot a dat na osách v grafu.

5.2 Vytvoření zdroje dat pro propojení programu Grapher s Databází

Propojení programu Grapher s databází realizuji vytvořeným zdrojem dat, který odkazuje na ovladač ODBC. Ve správci zdrojů dat ODBC, v záložce souborové DSN (Data Source Name) je možnost přidat nový zdroj dat Obr. 11.



Obr. 11 – Nový zdroj dat

Pro vytvoření spojení jsem použil Microsoft Access Driver (*.mdb) a uložil ho, pod názvem FILE.dsn do složky Data Sources (Zdroje dat).

5.3 Načtení dat z databáze

Skript by měl načíst celou databázi do vizuálního rozhraní programu Grapher, skript slouží pouze pro kontrolu toho, jaké záznamy se nachází v databázových tabulkách.

Skript, který načte celou databázi, tudíž všechny databázové tabulky z databáze Tunel Bedřichov.mdb pomocí funkcí, které obsahuje Scripter.

```
Set Worksheet2 = Grapher.Documents.Add(grfWksDoc)

Worksheet2.LoadDB(1,1,"Provider=MSDASQL.1;Persist Security
Info=False;Extended Properties=DBQ=C:\Grapher\Tunel Bedři
```

```
chov.mdb;DefaultDir=C:\;Driver={Driver do Microsoft Access
(*.mdb)};DriverId=25;Exclusive=1;FIL=MS Access;FILE
DSN=C:\Grapher\File.dsn;MaxBufferSize=2048;MaxScanRows=8;Pa
geTimeout=5;ReadOnly=0;SafeTransactions=0;Threads=3;UID=adm
in;UserCommitSync=Yes;","SELECT * FROM [senzor]")
```

`Příkaz Set Worksheet2 = Grapher.Documents.Add(grfWksDoc)` nastavuje ve skriptu konkrétní instanci worksheetu, je pojmenován jako Worksheet2 a přidává ho do Tabbed Windows v programu Grapher.

Příkaz `Worksheet2.LoadDB` vyvolává metodu načtení databáze a naplnění vytvořeného worksheetu. Čísla 1,1 znamenají pozici, odkud se worksheet začíná plnit. Hodnoty 1,1 odpovídají sloupci A a řádku 1. *Provider* značí poskytovatele spojení, zde to je MSDASQL.1. Dále je zde absolutní cesta k databázi `C:\Grapher\Tunel Bedřichov.mdb. Driver={Driver do Microsoft Access (*.mdb)}` značí ovladač připojení. FILEDSN=C:\Grapher\File.dsn cesta ke zdroji dat ODBC.
`SELECT * FROM [senzor]` příkaz v jazyku SQL značí výběr všech záznamů z tabulky data. SQL příkazy měním podle potřeby stahovaných záznamů z databáze. Například pro výběr hodnot a časů z tabulky data by příkaz vypadal takto: `SELECT hodnota, time FROM [data].`

5.4 Načtení dat z konkrétního senzoru a vytvoření grafu

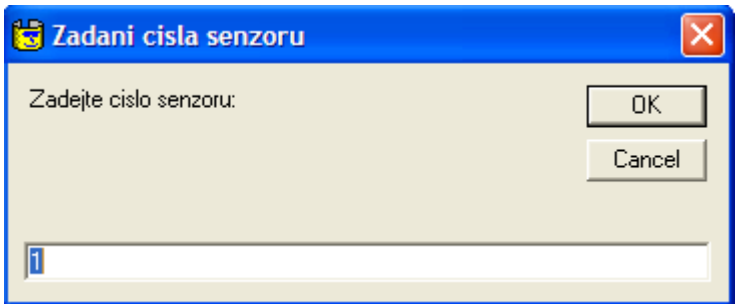
Skript je vytvořený pro uživatele, který bude vědět z jakého id senzoru si chce nechat zobrazit graf.

Pro uživatelský vstup používám funkci Inputbox.

```
Dim senzor

senzor = InputBox$("Zadejte cislo senzoru:", _ "Zadani
cisla senzoru","1")
```

`Dim senzor` vytvoří proměnou senzor, do které přiřazuji InputBox s Titulkem „Zadani cisla senzoru“, textem „Zadejte cislo senzoru“ a přednastavenou hodnotou „1“.



Obr. 12 - Inputbox

V proměnné `senzor` pak zůstane uložená hodnota z uživatelského vstupu. Proměnnou `senzor` později ověřuji, jestli uživatel nezadal nějaký nežádoucí text, například slovo. Ověřuji ho funkcí *IsNumeric(expr)* kde expr je kontrolovaná hodnota.

Po načtení zadané hodnoty od uživatele posílám dotaz do databáze popsané v kapitole 4.3. SQL příkaz vypadá takto: `SELECT time, hodnota FROM [data]`. Načtená data ukládám do worksheetu na Obr. 13

Plot1 | Worksheet1 | data.bln | Graf_tunel

A1: time

	A	B	C	D	E	F	G	H
1	time	hodnota						
2	5/1/2012 5:23:42 PM	10						
3	5/1/2012 5:23:52 PM	12						
4	5/1/2012 5:24:02 PM	11						
5	5/1/2012 5:24:12 PM	20						
6	5/1/2012 5:24:22 PM	22						
7	5/1/2012 5:24:32 PM	19						
8	5/1/2012 5:24:42 PM	16						
9	5/1/2012 5:24:52 PM	18						
10	5/1/2012 5:25:02 PM	20						

Obr. 13 – načtený worksheet

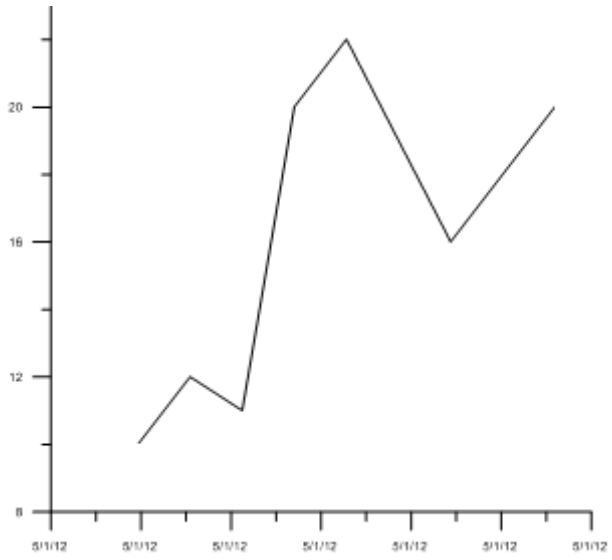
Vytvořený worksheet ukládám jeho metodou `SaveAs("C:\Grapehr\data.bln","ClipboardTextType=-1;Delimiter=comma;ExportCodepage=0;ExportEncodingMethod=0;TextQualifier=doublequote",wksFileFormatBln)`, kde je zadaná absolutní cesta k uložení worksheetu a formát ve kterém se ukládá. Worksheet ukládám pod názvem data.bln (Golden Software Blanking file format).

Po uložení worksheetu vytvořím v Grapheru nový graf s názvem graf_tunel `Set Graf_tunel = Grapher.Documents.Add(grfPlotDoc)`. Příkaz přidá do Tabbed Windows nový Grapher dokument. Graf naplním příkazem

`Graf_tunel.Shapes.AddLinePlotGraph("C:\Grapher\data.bln",1, 2)` kde načítám uložený soubor data.bln, ve kterém jsou uloženy načtené záznamy z databáze. Dále nastavuji konkrétní graf příkazem `Set Graph1 = Graf_tunel.Shapes.Item(1)`, čímž vytvářím instanci grafu.

Příkazem `Set YAxis1 = Graph1.Axes.Item(2)` nastavuji grafu 2 osy.

A jako poslední výsledný graf uložím `Graf_tunel.SaveAs("C:\Grapher\Graf_tunel.grf",grfStandard)` jako Graf_tunel.grf. Výsledný graf vidíme na Obr. 14:



Obr. 14 – výsledný graf pro načtení ze senzoru

5.5 Skript pro zobrazení grafů z více senzorů

Požadavek na skript je, aby zobrazil více čar v grafu. Skript by se měl připojit do databáze a nabídnout uživateli informace o jednotlivých senzorech, tudíž jejich jména (pracovní označení), na kterém zařízení se nacházejí, jakou veličinu měří a v jakém bodě jsou. Data načtená z databáze by se měla uložit do worksheetu pro vybraný senzor. Při vytváření více čar v grafu je nutno, jednotlivé čáry od sebe barevně rozlišit.

Ihned po spuštění skriptu se připojí k databázi a načte prvních 1000 záznamů do worksheetu. Na nalezení všech senzorů by mi mohlo 1000 prvních záznamů stačit, protože dle XML souboru, který přidávám v příloze, záznamy nejsou nijak řazeny. Senzory mohou posílat data v rozmezí minut i hodin. Původně jsem zamýšlel procházet prvních 100 záznamů, jenže bych nemusel dojít až k záznamům ze senzorů, co posílají

data v rozmezí hodin. Nebylo by od věci procházet celý načtený worksheet, ale když se do něj načte databáze, která má přes 20tisíc podobných záznamů, bylo by to časově náročné.

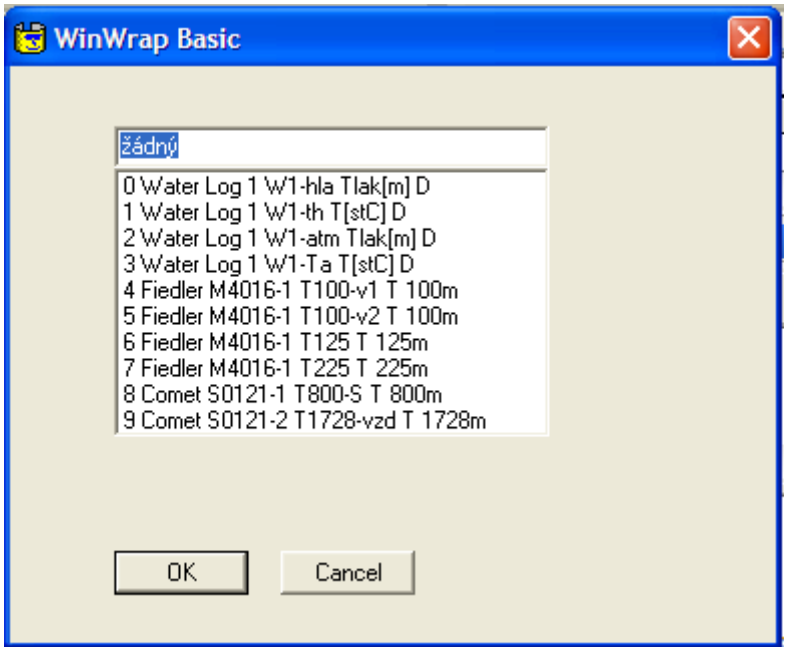
Worksheet poté prochází cyklem `while`, ve kterém vybírá ID_senzor a zjišťuje, jestli je již tento senzor uložen v mnou vytvořeném komboboxu, jestli ne, tak ho do něj přiřadí i s jeho pracovním názvem, polohou, zařízení na kterém je umístěn a veličinou. Následně skript spustí uživatelský dialog, kde si uživatel vybere, jaký senzor si chce nechat zobrazit. Když si nevybere žádný, tak se skript ukončí. V opačném případě se načtou hodnoty vybraného senzoru do grafu. Jakmile je načten, znovu se zobrazí uživatelský dialog s možností výběru grafu. Uživatelský dialog se zobrazuje tolikrát, kolik je načtených senzorů v mém komboboxu. Grafy se načítají postupně do jednoho grafického zobrazení grafu barevně odlišeny.

Samozřejmě na začátku skriptu se musí nastavit Grapher jako objekt a vytvořit jeho instanci. Stejně jako v kapitole 2.4.1. Skript si načte záznamy z databáze do pomocného worksheetu, který pojmenuje `pomocny`. Dále si vytvoří kombobox o velikosti 1000, `Dim combos$(1000)a` dvě proměnné `x = 1` a `y = 2`, tyto dvě proměnné jsou proto, když skript prochází načtený worksheet začíná od druhé pozice prvního sloupce, což značí sloupec o hodnotách ID_senzor. (na pozici 1,1 je název sloupce načtený z databáze, v tomto případě ID_senzor). V cyklu `While y < 1002` si vytvoří proměnnou senzor a přiřadí do ní hodnotu z worksheetu `pomocny.GetWksCell(y, x)` a poté inkrementuje `y=y+1`. Tímto cyklem projde 1000 záznamů v načteném worksheetu a nalezne tak id senzorů, které předá do uživatelského dialogu.

Další kombobox o velikosti 30 záznamů, který vytváří příkazem `Dim kombobox$(30)` už bude obsahovat všechny informace načtené z databáze, proto může být načten v uživatelském dialogu na Obr. 15. Další vytvořené proměnné jsou `device=“”, pracovni_nazev=“”, velicina=“”, bod_mereni=“”` (všechno jsou proměnné typu String). Pro naplnění komboboxu procházím kombobox `combos` dokud nenajdu první záznam ve worksheetu (ID_senzor) a když najdu shodu, příkazem `Join(Array(senzor, pracovni_nazev, device, velicina, bod_mereni))` přidám nově vytvořený String do komboboxu `kombobox`.

Další částí je už uživatelský vstup, čili zobrazení uživatelského dialogu. Na Obr. 15 vidíme, jak vypadá uživatelský vstup do skriptu. Kde si uživatel může vybrat ze senzorů, které jsou uložené v komboboxu. Například ze zařízení Water Log, Fiedler M4016–1 nebo Comet S0121-1/2 v různých bodech od 100m do 1728, které měří teplotu.

Možnost „žádný“ slouží pro ukončení skriptu pomocí funkce `If dlg1.combo$ = "žádný" Then Exit All`.



Obr. 15 - Uživatelský dialog

Níže je popsáno, jak vypadá Uživatelský dialog ve skriptu:

```
Begin Dialog UserDialog 510,266
          Text 100,100,180,15,"Please push the OK button"
          ComboBox 70,28,290,154,kombobox$(),.combo$
          OKButton 70,224,90,21
          CancelButton 180,224,90,21
End Dialog
Dim dlg1 As UserDialog
dlg1.combo$ = "žádný"
Dialog dlg1 ' show dialog (wait for ok)
```

Jako základní hodnota je označená hodnota žádný. Dialog se spustí a čeká, jaký senzor si uživatel vybere.

Po vybrání jednoho z prvků komboboxu (senzoru) se vytvoří worksheet `Set Wksheetsenzoru = Grapher.Documents.Add(grfWksDoc)` a přidá se do Tabbed Windows. Jelikož jsou některá data hodně malá (0,0000213), musím ve sloupci B, kam se načítají hodnoty z databáze nastavit počet desetinných míst a typ, který by měl očekávat, příkazy

```
Wksheetsenzoru.Columns("B:B").Format.NumericType = wksType Exponential
Wksheetsenzoru.Columns("B:B").Format.Digits = 15
```

První příkaz nastavuje sloupec na exponenciální data a druhý dává sloupci možnosti až 15 míst za desetinnou čárkou.

Poté příkazem `dodb = Left(dlg1.combo$, 1)` vybírá první dvě místa z vybraného komboboxu. Tím vybere číslo senzoru.

Číslo senzoru skript předá funkci `Wksheetsenzoru.LoadDB(?Row, ?Col, ?Connect, ?SQL)`, která načte z databáze do worksheetu čas a hodnotu.

Worksheet ukládám na absolutní pozici `Wksheetsenzoru.SaveAs("C:\Grapher\Wksheetsenzorzu.bln","ClipboardTextType=-1;Delimiter=comma;ExportCodepage=0;ExportEncodingMethod=0;TextQualifier=doublequote",wksFileFormatBln)`

Když uživatel vybírá první senzor, skript nejdříve vytvoří graf, v případech přidávání senzorů do grafu se už nově načtené data ze senzorů pouze přidávají do již vytvořeného grafu.

Přidání do Tabbed Windows provede `Set Graf_tunel = Grapher.Documents.Add(grfPlotDoc)`, další načtení hodnot z uloženého worksheetu zajistí `Graf_tunel.Shapes.AddLinePlotGraph("C:\Grapher\Wksheetsenzorzu.bln",1,2)`. Příkazem `LineScatterPlot1.line.foreColor = grfColorRed` jsem nastavil čáře grafu červenou barvu. Barvy měním podle počtu přidaných senzorů.

Pro přidávání dalších grafů už jsem používal příkaz `Graph1.AddLinePlot("C:\Grapher\Wksheetsenzorzu1.bln",1,2,"X Axis 1","Y Axis 1")`, který vložil do grafu, do stávajících os X a Y, graf vytvořený z worksheetu, který patří dalšímu načtenému senzoru. Příkazem `Set LineScatterPlot2 = Graph1.Plots.Item(2)` jsem vytvořil druhý

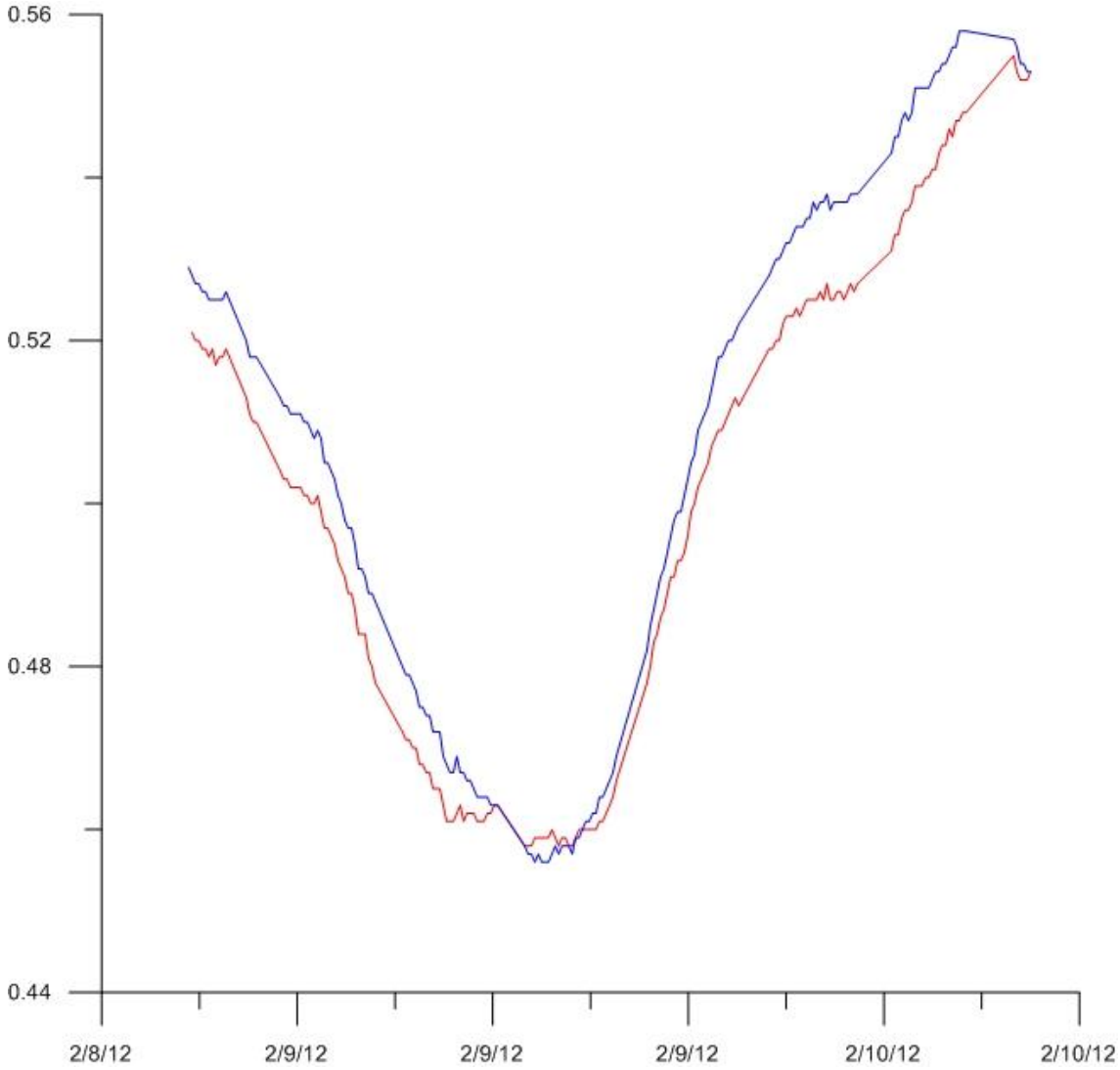
LineScatterPlot do `Graph1` a nastavil jsem jeho barvu na modrou. Takto měním barvy podle počtu načtených senzorů.

Skript je omezený na určitý počet čar zobrazených v grafu. Při jeho vytváření jsem se v programu Grapher setkal s jistým omezením. Pro vytvoření grafu, se načítá uložený worksheet, bohužel, worksheet nejde uložit nebo přepsat, když je již v programu Grapher otevřen. Zůstávají v něm stejné hodnoty, které do něj byly vloženy při prvním načtení. Proto pro každý přidaný senzor, vytvářím nový worksheet. Každý nový worksheet pod jiným jménem Wksheetsenzoru až Wksheetsenzoru9. Nemělo by se tak přihodit, že budu mít 2 stejně načtené worksheety pro 2 různé senzory. Takže maximální počet čar a tím pádem počet zobrazených senzorů v grafu je 10.

V datech, se kterými jsem pracoval, se stávalo, že k časovému údaji scházela naměřená hodnota. Grapher ji vyhodnotil jako nulovou. V nenulové řadě naměřených hodnot, pak v grafu čára klesá na nulu a poté zase stoupá a pokračuje v řadě. Pro úplnou automatizaci, bych doporučil funkci, která projde načtené data a záznamy a ty, ve kterých není žádná hodnota tak vymaže. Viz Obr. 18

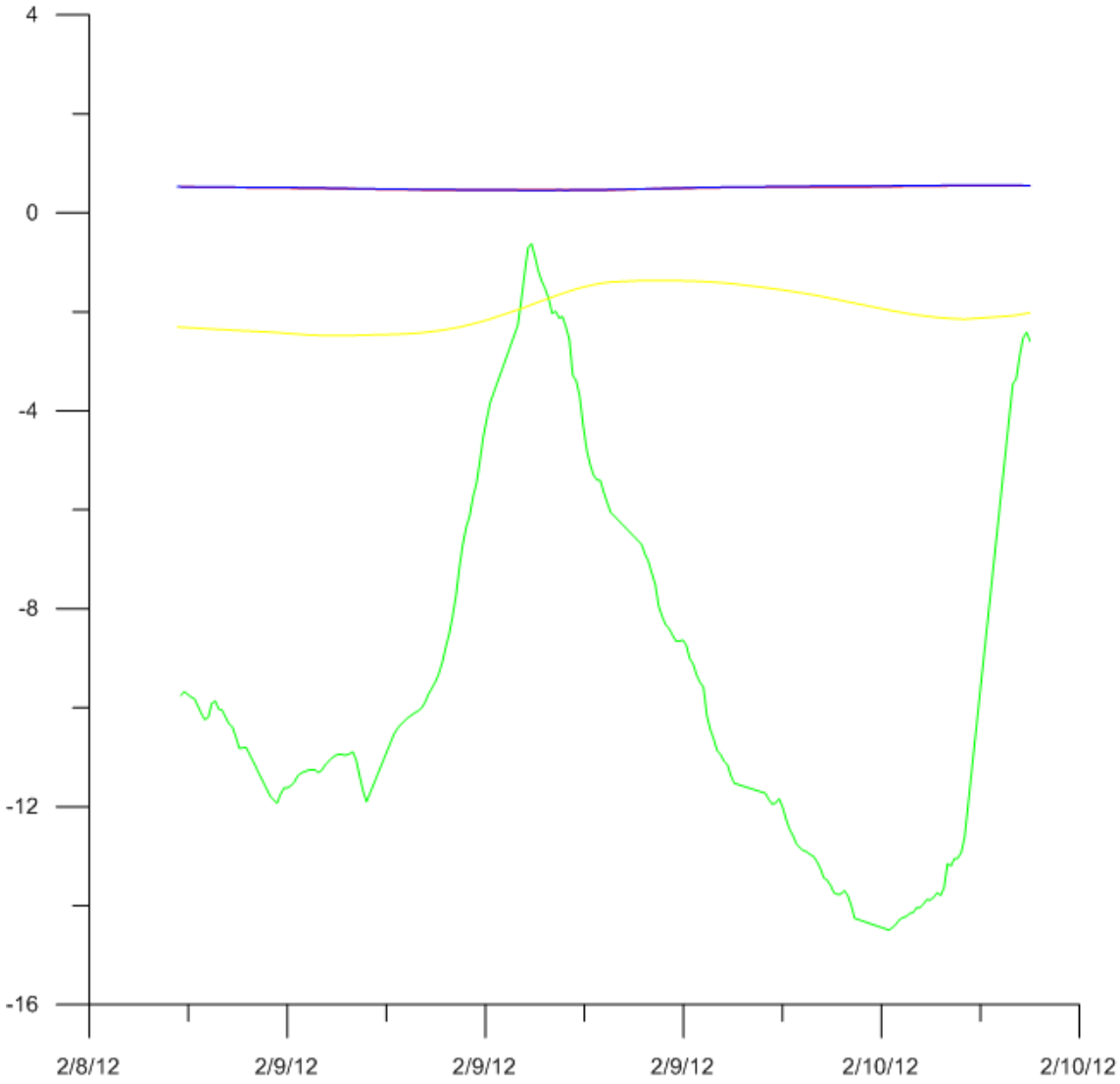
Při zobrazení více senzorů je dobré vědět, v jakém rozsahu porovnávané senzory zobrazujeme. Když budou hodnoty velice rozdílné, například jeden senzor bude měřit hodnoty v rozsahu 0,40 až 0,60 a druhý v rozsahu 0 až 20, bude velice k nerozeznání jejich různost. Nebo při porovnávání v jiných časových intervalech. Čáry by se nemusely v grafu sejít, případně by to data z jednoho senzoru zobrazilo v zanedbatelném intervalu. Například záznamy ze senzoru, který odesílá data po minutě v podobě 100 vzorků, nebude vhodné porovnávat se 100 vzorky senzoru, který odešle data jednou za den.

Aby uživatel mohl pracovat se skripty. Musí si stáhnout složku „Grapher“ z přiloženého CD, která obsahuje skripty, zdroj dat ODBC a vytvořenou databázi na disk C. Pro možnost úpravy databáze je potřeba Microsoft Access. Skripty jsou navrženy na práci s absolutní cestou, takže kdyby se soubory nenacházely na adrese C:\Grapher\ nebylo by možné se skripty pracovat.



Obr. 16 - Graf porovnání dvou čar v grafu.

Na Obr. 16 vidíme výsledný graf po načtení ze senzorů (červený) „0, water Log 1, W1-hla, Tlak[m], D“ a (modrý) „2, water Log 1, W1-atm, Tlak[m], D“. V databázi bylo přes 700 záznamů k jednotlivým senzorům. Graf se zobrazuje s hodnotami od 0,45 až po 0,55 v době od 8.2.2012 až do 10.2.2012, kdy se záznamy do databáze ukládaly po deseti minutách.

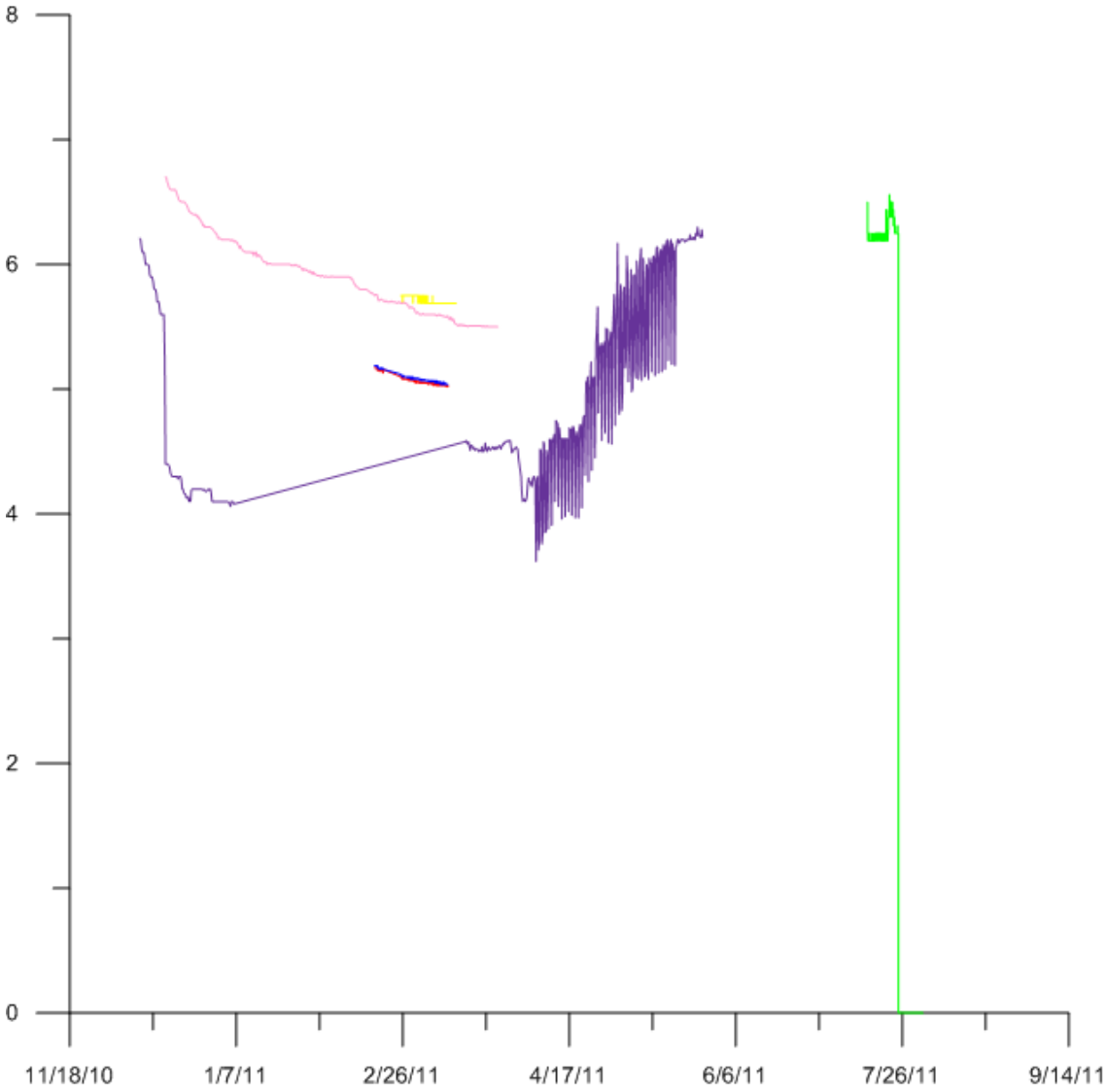


Obr. 17 - Graf ze srovnání 4 čar, data s malým rozsahem hodnot s daty s velkým rozsahem hodnot.

Obr. 17 nám zobrazuje 4 grafy ze 4 senzorů.

- (červený) „0, water Log 1, W1-hla, Tlak[m], D“,
- (modrý) „2, water Log 1, W1-atm, Tlak[m], D“,
- (zelený) „1, water Log 1, W1-th, T[stC], D“,
- (žlutý) „3, water Log 1, W1-ta, T[stC], D“.

Stejně jako u předchozího obrázku platí počet záznamů v databázi přes 700 a čas 8.2.2012 až do 10.2.2012 po 10 minutách. Tlaky, ze senzorů 0 a 2 (z předchozího obrázku) v tomto grafu budou těžce viditelné díky rozsahu od 0,45 až po 0,55. Tento graf má rozsah -15 až 1. Je tu viditelné porovnání teplot ze senzorů 2 a 3 v rozmezí 2 dnů.



Obr. 18 – Zobrazení 6 čar v rozdílných časových údajích

Obr. 18 nám zobrazuje měření z dvou zařízení, Fiedler M4016-1 a Comet S0121-1/2. Na obrázku je zobrazeno 6 grafů. Modrá a červená zobrazuje teploty v kanálu na 100 metrech. Zelená zobrazuje teplotu pramenu ve 125 metrech, svislá čára dolů značí, že v databázi už byly nulové hodnoty. Žlutá zobrazuje teplotu pramenu v 225 metrech. Růžová zobrazuje teplotu uvnitř skály v 800 metrech. Fialová zobrazuje teplotu vody uvnitř hadice v 1700 metrech.

Na obrázku se zobrazuji data měřená v různých časech s velice podobnou hodnotou. Záznamy pro jednotlivé senzory se pohybují okolo 300.

6 Závěr

V bakalářské práci jsem navrhl skripty, které automatizují práci s programem Grapher. Skripty se pomocí databázového rozhraní připojují k databázi a stahují si z ní data, ze kterých automatizují tvorbu grafu. Skripty dokážou načíst obsah databáze, zobrazovat jednotlivé grafy a porovnávat data z více senzorů v jednom grafu. Při porovnávání dat z více senzorů, jsou pak čáry barevně odlišeny a počet zobrazených čar je maximálně 10.

Původní záměr práce, byl připojit se k živé databázi tunelu Bedřichov, ale kvůli technickým omezení programu Grapher, jsem musel demonstrovat připojení k databázi na programu MS Access.

Během tvorby práce jsem si tedy musel vytvořit vlastní databázi, kterou jsem plnil daty z databáze tunelu Bedřichov, protože program Grapher mi neumožnil připojit se do databáze na serveru v Liberci. Vlastní vytvořená databáze, ze které skripty stahují data je totožná s databází tunelu Bedřichov, která je na serveru.

Během zpracování práce se ukázalo omezení při používání skriptů. Skripty je možné spouštět pouze na 32 bitových operačních systémech. 64bitové verze systému nepodporují vytvořené databázové rozhraní.

Seznam použité literatury

[1] *AccuTerm 2K2: Scripting Language Reference Manual.* Sunland, CA, USA: AccuSoft Enterprises, 2003.

[2] Bedřichovský tunel. [online]. Liberec: Technická univerzita v Liberci. [cit. 15. 11. 2011] URL: <http://bedrichov.tul.cz/Tunel/index.php>

[3] BĚLOCH, M. *Programy Grapher, MS Excel a MatLab a možnosti jejich připojení k databázi prostřednictvím rozhraní ODBC.* [projekt] Liberec, Technická univerzita v Liberci, 2012.

[4] *Databáze*. [online]. Wikipedia.org [cit. 29. 3. 2012] URL: <http://cs.wikipedia.org/wiki/databáze>

[5] *Grapher User's Guide, 2D & 3D Graphing Software for Scientists, Engineers & Business Professionals*. Golden, CO, USA: Golden Software, Inc., 2011.

[6] HOKR, M. a kol. *Tunel Bedřichov: Charakterizace granitoidů in situ.* [závěrečná zpráva]. Praha: SÚRAO, 2010.

[7] HOKR, M. a kol. *Zpráva pro 2. kontrolní den projektu TUNEL 2011*. Liberec: Technická univerzita v Liberci, 2011.

[8] MUKNŠNÁBL, J. *ODBC v kostce.* [online]. Reboot.cz [cit. 30. 4. 2012] URL: <http://reboot.cz/howto/databaze/odbc-v-kostce/articles.html?id=152>

[9] *Relační databáze*. [online]. Wikipedia.org [cit. 29. 3. 2012] URL: <http://cs.wikipedia.org/wiki/Relační_databáze>

[10] SVOBODA, P. *Zařízení pro vzdálený sběr a přenos dat – Firmware.* Liberec: Technická univerzita v Liberci, 2011.

Seznam příloh

Příloha A – Seznam použitých měřících zařízení v Tunelu Bedřichov

Pramen	Automatické měření, frekvence záznamu				Popis
	výtok	Teplota	pH	redox	
AV1	sklopka Fiedler 5 min	…	…	…	žlábek v metráži 125 m od portálu
AV2	Lapač srážek se sklopkou MicroLog T3 každé sklopení	Teplotní čidla Comet 10 min	…	…	dva žlábky v metráži 798,5 m (vzdálenější má č. V 2/1 blíže k portálu a o něco níže ve stěně je V 2/2). Žlábky jsou na dvou různých drobných puklinách ve vzdálenosti 30 cm
AV3	Sklopka Comet 10 min	…	…	…	žlábek umístěný na metráži 1375,5 m
AV4-1		Teplotní čidla Comet 15 min	…	…	dvě hadice v místech zabetonovaných puklin ve stropě v pravém boku tunelu A v metráži 1728,5 m. V4/1 je umístěna výše ve stropě a V4/2 o něco níže
AV4-2	Hladinoměr Levelogger 5 min	…	…	…	
AV5	sklopka Modul infrastr. 5 min	…	Sonda theta90 Modul in 5 min	…	ze stěny tryskající pramen, zjištěný je v roce 2006 v metráži 226,4 m. V roce 2007 poklesu jeho výtoku byl opatřen žlábkem
AV6	Sklopka Modul infrastr. 5 min	…	…	…	výtok ze starší železné roury v pravém boku v metráži 142 m
AV7	hladinoměr Levelogger 15 min	…	…	…	hadice v metráži 76,5 m, je to čtvrtá hadice od portálu z pěti, která má vždy nejsilnější výtok z okolních hadic
AV8	…	…	…	…	celkový výtok z potrubí sbírající vodu z celého tunelu na u portálu tunelu A
AV10	…	…	…	…	výtok ze sběrného potrubí na metráži 2420m
W67	…	…	…	…	hadice v místě zabetonované pukliny v pravém boku tunelu na metráži 67 m
W142					hadička ústící z navrtané pukliny v pravém boku tunelu na metráži 142 m (sousedí s pramenem V6)
W1565					přehrazení skalní prohlubně v levém boku tunelu na metráži 1565 m, vývod realizován hadičkou; pro odběr vzorků bez přístupu vzduchu slouží další hadička zapuštěná dovnitř skály
W2210					do žlabu svedený výtok z upraveného otvoru v betonovém nástřiku, levý dolní bok tunelu na metráži 2210 m
W2313					hadice v místě zabetonované pukliny na metráži 2313 m v pravém boku tunelu
W2470	Hladinoměr Levelogger 15 min				výtok ze staré železné roury v levém boku tunelu na metráži 2470 m
Kanál 2470m	…	Teplotní čidla Comet 15 min	… …	… …	
Kanál 100m	Výška hladiny, ultrazvukové čidlo US1200 Fiedler 5 min	Teplotní čidlo Fiedler 5 min	Čidlo pH Fiedler 5 min	Čidlo ORP Fiedler 5 min	

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: **2011/2012**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal Běloch**
Osobní číslo:	**M08000121**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Automatizace tvorby grafů v programu Grapher využívající rozhraní k databázi**
Zadávající katedra:	**Ústav nových technologií a aplikované informatiky**

Zásady pro vypracování:

1. Seznamte se se softwarem Grapher (Golden Sofwtare) pro tvorbu vědeckých grafů a možnostmi pokročilé konfigurace a programování (Visual basic skripty, rozhraní ODBC).
2. Navrhněte metodiku pro zpracování grafů z průběžně rozšiřované časové řady měření (struktura vstupního souboru, formát časového údaje, automatický update změněného souboru do definovaného vzhledu grafu).
3. Otestujte řešení na různých typech dat a ošetřete jednotnost vzhledu např. pro různě hustá data a více rozsahů os.
4. Předveďte použití rozhranní ODBC pro načítání dat pro grafy z databáze.